D.R.O.W.N.

Description: Padre and I discuss the week's major security events, including the FBI's hearing delay, Matthew Green's iMessage attack, a side-channel attack on phones, a massive malvertising campaign affecting many major sites, the 2016 Pwn2Own contest, a new Android Stagefright vulnerability and attack, and some other miscellany. We then describe the DROWN attack against up-to-date TLS servers using still-present SSLv2 protocol.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-552.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-552-lq.mp3

SHOW TEASE: The FBI has decided that they don't need Apple. Or do they? Plus Lenovo goes further down the malware drain. iMessage is broken. Stagefright is the gift that you don't want that keeps on giving, and the Internet is trying to DROWN you. It's all coming up next on Security Now!.

FATHER ROBERT BALLECER: This is Security Now!, Episode 552, recorded March 22nd, 2016: D.R.O.W.N.

It's time for Security Now!, the podcast all about security with the man himself, that's right, it's the Minister of Malware, the Bishop of Bytes, Mr. Steve Gibson. Of course Steve Gibson from GRC.com. He is the purveyor, the inventor, and the originator of the security and the tools that you need. Steve.

Steve Gibson: The Minister of Malware.

PADRE: I like it, yeah.

Steve: I'm not sure about that one.

PADRE: Well, you know, it's sort of the...

Steve: Maybe the coiner of the term "spyware." I like that one better than the Minister of Malware.

PADRE: Yeah, actually, yeah, Minister of Malware has some sinister tones to it.

Steve: Yeah, that's a black hat sort of designation. And of course everyone listening suddenly realizes that you're not Leo.

PADRE: No, I am not Leo Laporte. I'm Father Robert Ballecer, known as the Digital Jesuit. I'm in for Leo Laporte who is on a trip to the East Coast, we hope. That's assuming that he's making his flight on time. It's probably going to be a bit busy at the airports right now with everything that's happening in the world.

Steve: Yeah, he's probably on that little dirt path that connects Petaluma to San Francisco that nobody has bothered to, like, deal with for the last two decades.

PADRE: Oh, it's fantastic, though, I love it because every time it rains there's a little lake that forms between San Francisco and Petaluma, and you have to navigate through it.

Steve: It makes the ducks happy. And, you know...

PADRE: Very happy, indeed.

Steve: ...that's important. So, as promised, we're going to talk about DROWN is our main topic this week, although we've had lots of fun and interesting news that we have, of course, we'll talk about the FBI postponing the big hearing versus Apple, for those who are, like, out of the news and out of social media and might somehow not know that that happened yesterday. Matthew Green, our cryptographer friend at Johns Hopkins, and four students managed to find a crack in iMessage. And there's some really interesting takeaways from that.

We've got a side-channel attack which we've been talking about a lot lately. We covered one a couple weeks ago that involved pulling crypto keys from a laptop on the other side of a wall. In this case, it's using mobile devices and just some stuff you can get, I was going to say at Radio Shack, but I don't think you can get anything at Radio Shack any longer. But it doesn't require any high-tech stuff. You use a soundcard and a coil of wire that you wind yourself. So that's a little problematical.

Then there's been a big malvertising campaign, speaking of malvertising, or malware. Lenovo back in the doghouse. We had the results from the 2016 Pwn2Own competition, which had some interesting outcomes. Stagefright is back again. And although it's affecting a smaller percentage of Android devices, it's affecting them to a much more reliable, if you can use that term, like in terms of the reliability of the bad guys to do a drive-by code execution. So we'll get into the details of that.

A bunch of people have asked me, because I'm building, or have built, a big new replacement desktop system because I heard that - it was like about a month ago, I'm sure you know, Padre, that Microsoft announced that the Skylake family of processors moving forward would not bother to be supporting Windows 7. Well, since I certainly am supporting Windows 7 and have no interest in going forward, I thought, oh, crap, I have to make a Haswell-based system now because I can't wait for five years until I would maybe otherwise do it. So I was talking a little bit about the configuration of that, and I had some questions from people who wanted to know some details. So I thought I'd fill them in on a little bit of that.

We have some other miscellaneous stuff. And then this DROWN attack, I was thinking a couple weeks ago we were going to talk about DROWN and something else. There were two things, and I don't remember what the other one was off the top of my head. But they both ended up being so big and interesting, and there was so much news that week, that I thought, okay, we can't squeeze both of these in.

So this gets its own episode, which I think has got another real fundamental crypto

teachable moment because this is a way for our state-of-the-art latest and greatest transport layer security, TLS server-protected protocol, or server-protecting protocol, to be undermined if the old, creaky, deprecated, nobody actually uses it anymore, SSLv2 happens to still be around, even if you're not using it. So it's a very clever attack where the presence of this deprecated, unused SSL protocol can actually give attackers a leg up in attacking the actual protocol that clients and servers use. So I think just Security Now! Episode 552 is going to have something for everybody, and a great one.

PADRE: Yeah, it's going to be absolutely packed. And I've got a lot of questions for you on DROWN because last week, when you first sent me the link, and you said, "This is what I'd like to talk about," first I did all my own research. And then I started talking to a couple of my network security friends. And again, we'll talk about this when we get to the end of the show. But they said something to the effect of this is the new advanced persistent threat. Not necessarily something that has actively been put into your system, but deprecated software and software modules that don't completely remove themselves from services. As we have servers that age out, they tend to accumulate stuff that you don't even know is there.

Steve: Yup.

PADRE: And so more and more of them, their compliance tests actually have to do with saying, when was the OS installed? Because if it was installed past a certain date, there's a good chance you have things that you don't know are still in the system. So it's very interesting, yeah.

Steve: So there's absolutely that, which is what I was referring to, actually, as this item of teachability. And the other is, and we'll get to it, is that they also found a way of leveraging a problem that had been fixed in OpenSSL in March of 2015. So one year ago. And it turns out that 24% of running servers now are still using an OpenSSL earlier than a year ago. And we know how many problems have been fixed since then. They're not crucial and oh, my god, run around, like Heartbleed was, for example. But it's demonstrating that, unless there's some crucial need, I mean, one of the other things we keep looking at and addressing on the podcast is that it just takes, you know, acts of, well, since I'm here with you, God to move security forward.

And so, if there isn't something that is incredibly, crucially, horribly broken, servers don't get updated. And here's a quarter of the servers running an OpenSSL for their security that is a year old. And in this case, this attack allows, instead of this thing taking, like, GPU-class attack 18 hours, it can be done in almost, well, actually it can be done in real-time on a single laptop if the server has an OpenSSL version more than a year old. So it's like, okay. I mean, we keep coming back to these things. Nobody wants to change what's working, even though what works and what is secure are really different things.

PADRE: It's the whole, it's working right now, and there's a chance it won't hit me. That's the sort of mentality that I see an awful lot.

Steve: Yeah.

PADRE: Steve, let's bring it on in. What do we start with?

Steve: So we don't know a lot about what's actually going on with the FBI. Of course everybody knows of the big fight that the FBI and Apple have been having, that the FBI is trying to use an ancient law, the All Writs Act from the 1700s, in order to compel Apple to modify, create a modified version of iOS which will essentially remove the anti-brute-

forcing limiters from the software and also provide a new means for the FBI to input guesses of the passcode that the San Bernardino bomber or, not bomber, gunman Farook used. And of course Apple is resisting that, saying that this is a slippery slope, and it's not just one phone, it's hundreds, and so forth.

So yesterday morning, during Apple's spring presentation where they introduced the new phone and the smaller iPad Pro and so forth, Tim Cook began by talking about tomorrow's hearing, meaning that yesterday morning this was on. And then suddenly, in the afternoon yesterday, we hear that the hearing has been at least postponed. What happened was that federal prosecutors yesterday filed saying that on Sunday, March 20th, so that's day before yesterday, an outside party demonstrated to the FBI a possible method for unlocking Farook's phone. Testing is required to determine whether it is a viable method that will not compromise data on Farook's phone. If the method is viable, it should eliminate the need for the assistance from Apple set forth in the All Writs Act order in this case.

And this generated a social media Twitter storm. Jennifer Granick, who's a well-known Director of Civil Liberties at the Center for Internet and Society at Stanford Law School, she tweeted: "We don't know whether the government withdraws its motion" - or, I'm sorry, "We won't know" - I quoted this deliberately, I mean, I copied and pasted this, so this must - she said, "We won't know whether the government withdraws its motion until at least April 15th."

PADRE: Right.

Steve: So I guess this is just a postponement pending formal withdrawal - I'm sorry, on April 5th.

PADRE: Yeah. That's the deadline because now that they've asked for this postponement they have until April 5th to either continue or to drop it.

Steve: And so what's so odd about this is, I mean, the conspiracy theorist who sort of suggests maybe we're not - what we're seeing is the surface of what's happening, and there's a lot more going on underneath. Because the operating theory was that the FBI almost certainly knew there was nothing actually on this phone. They did have the iCloud backup from five weeks before, so they were able to assess how actively used it was. This wasn't the phone that Farook bothered to physically destroy. He did destroy his own, so one presumes that's the one that actually had something useful on it, and it's destroyed beyond recovery.

So this whole thing sort of smelled a little fishy, like the FBI had chosen this to take a stand on because of their position. And then didn't we hear, I think it was Monday, that they had also - they were going to introduce some witnesses. And people who were reading this whole thing from a distance said, you know, that sort speaks to them not having a really strong position, if they're going to be bringing witnesses in, in order to testify as part of this. So this whole, oh, yesterday someone showed us a way to do this. And of course then the jokes were, oh, I wonder if John McAfee gave them a call.

PADRE: And of course remember John McAfee at first suggested social engineering, which works really well on dead people is what I've heard. And then he said, no, no, that was a joke, and geeks understood the joke. And what he actually would do, and he suggested the process of using some acid and a laser to try to get the unique hardware ID number off of the inside of the phone, which they could then hash against a PIN and, yeah, it was kind of silly.

Steve: Yeah, loony tunes.

PADRE: And I know a lot of people, and again the FBI has not been forthcoming about any process they're actually trying or who their...

Steve: Right, they've said nothing.

PADRE: Said nothing. But it doesn't sound like it's going to be the acid and laser issue because you don't test that. Either it works or doesn't work.

Steve: Some have speculated that it'll involve dismounting the nonvolatile ROM and cloning it because it could be removed and cloned. And that, then, creates a snapshot, essentially, of the uncompromised phone. Then maybe they substitute in RAM instead of the nonvolatile ROM. So now they have something that the phone can't tell it's not nonvolatile ROM. It thinks it's talking to the ROM. They do their guesses. It locks. Then they just re-update the RAM from their snapshot that they originally took, which essentially allows them to dynamically reset the state of the phone after every 10 mistaken guesses.

PADRE: Let me ask you about that, Steve. So we know that the way that the iPhone protects itself is it has a key that it will destroy if too many incorrect guesses are made.

Steve: Correct.

PADRE: But where is that key stored? It's not stored in the same memory as the system memory; is it?

Steve: Apple's a little unclear. We assume it's in the Secure Enclave. And, I mean, the whole point of the Secure Enclave is that it is part of the processor where there is no API. There's no programmatic means of asking it to divulge any secrets. You can't, there's no way to say, "Give me your key." All you can do is say, "Use your key to sign this," or "Use your key to decrypt this." So you can get it to do the work for you, but you can't ever get it to release its information.

And so anyway, it was some dialogue that I saw that Matthew Green and some people were making yesterday, this whole notion of cloning the NVRAM in order to, like, get a snapshot of the state. But if in fact the symmetric key, the master symmetric key gets erased, proactively wiped, and if that's in the Secure Enclave, which is on the whatever it is, A8 processor, then I don't know how you...

PADRE: Yeah, you're out of luck.

Steve: ...prevent that from happening.

PADRE: Yeah, because you can't store the master key in the NAND because the master key is required to open up the NAND.

Steve: Correct.

PADRE: That's what's encrypting the NAND.

Steve: Exactly.

PADRE: So even if you encrypt the NAND, so that's not the part that would get destroyed. The part that gets destroyed is the master key. Which as far as I know is actually stored, like you said, in the A8 processor itself, which you cannot clone.

Steve: Right.

PADRE: So, I mean, I understand people like this idea of NAND cloning. But then you get back to the original problem, which is you're left with a ridiculously difficult thing to decrypt. It's no longer a PIN. You're looking for a 256-bit AES cipher.

Steve: Exactly. Exactly. And, I mean, maybe, well, no, I mean, that is too many bits even for the NSA, unless they actually do have a quantum computer in the basement that we've been paying for but haven't known, and that's why they have power feeds coming in from Utah and cooling lakes and so forth. Who knows? But it seems very unlikely. I mean, that is too many bits for anyone to brute-force. And as you say, the beauty or the possibility of brute-forcing is the fact that it is a, what, four, maybe six-digit PIN that, if you can do it fast enough, there just aren't that many combinations to try.

PADRE: Yeah. We've got "Greenbits" in the chatroom who's saying that cloning a NAND is just a safety measure. But it's an unnecessary safety measure because the NAND, the iPhone will not destroy the NAND.

Steve: Right.

PADRE: The iPhone will throw away the master key.

Steve: To the NAND, exactly.

PADRE: Whether or not you clone it, it's not going to be affected. But again, this all could just be a ruse. This could just be the FBI's way of backing out of this because they realized Apple's fighting it a lot harder than they thought they would, and they're not going to get the precedent out of it that they wanted. And they definitely don't want a reverse precedent. They don't want the court coming down and saying, no, Apple was right in this case, you can't force them to do this.

Steve: Well, and we also have this pending Burr-Feinstein bill, which is supposed to hit any day now. I've never tried setting up a news alert, but I did, you know, Google has the ability to set up alerts. And so I've got Google looking for me for any mention of Burr-Feinstein because this is the legislation which has been in the works since the Paris attacks. And Dianne Feinstein, the California senator, she's on the Intelligence Committee, she has been absolutely pro government ability to decrypt on demand. And, I mean, my guess is that the legislation will say that, after a certain date in the future to give companies time to come into compliance, it will be necessary for commercial companies to be able, however they choose or design, the government is not cryptographers, they're legislators and law enforcement.

And so like after January of 2018, or maybe it'll be '19 or '20, so the idea would be after some point it must be that commercial services are able, however they choose to arrange it, to comply with government orders to decrypt their customers' data. And, I mean, it could just be that. I'll be surprised if it isn't something like that. It can't be immediate. And I can't see that it's going to be any sort of a lesser request. I mean, that's what the government says it wants.

PADRE: It is a fantastic example of legislation without technical knowledge. You can read through the text of the bill and say, yeah, you didn't talk to a single engineer or mathematician, did you, because it's like one of these, well, this has to happen. And you ask them, well, how are you going to make that happen? I mean, you understand all the - no, no, no. You're just going to make it happen. And we get back to Apple, which is you're now going to the same technology companies that have provided features for their customers because of their customers' fear of digital snooping, and you're asking them to either backpedal on the security features that they've offered, or to actually break what they've designed to be unbreakable.

Steve: Right.

PADRE: Yay.

Steve: Right. And as we've talked about on this podcast, Apple, I mean, you can understand. Apple is selling security. And they want to produce a phone they cannot break. I mean, it's in their best interest to say to their customers, who like that as a bullet point, and also to the government, because they don't want a queue of phones that they have to crack open. They want to be able to say sorry, this is end-to-end encryption. We don't have the keys. And that just ends, it just ends it. And you know that there are people, I fact, when the story began to happen, there was word that Apple was at work on removing even this little wedge concept that the FBI came up with. I mean, we have to give, I give the FBI some props for cleverness in looking at the situation and saying, okay, what's the minimum that we need in order to do this?

And again, this is just taking it all on its face, assuming that this is not for the sake of media and creating a precedent and so forth. If the FBI actually did believe that this phone that had been backed up as recently as five weeks before - and who knows if it had even been used in any meaningful way since. If they really wanted to get in there, they came up with a clever solution. It's we need to remove a couple limiters, and then we need the ability to put guesses in faster. And then of course maybe the step too far is their need to compel Apple to do that. And of course in last week's reply to Apple's reply, in a footnote they said, well, if Apple doesn't want to do the work, give us your private key and source code, and we'll be happy to do that for you.

PADRE: Yeah.

Steve: Which of course no one ever believed would ever happen.

PADRE: It's not going to happen.

Steve: Anyway, as I said months ago here, this is the issue of our time. This is just a gripping, really interesting contest between math, which is now perfect enough to create unbreakable crypto, and the governments of the world's need or desire to see into that, to crack it anyway.

PADRE: Steve, I want to go off on a tangent. I know we don't do tangents on this show. But one of the things that has been interesting coming out of this was Microsoft has taken a different approach to government encroachment on digital security. They started up their Azure Deutschland service. And so what they've done is they've found a government that is incredibly tough on privacy and digital rights.

Steve: Germany.

PADRE: Germany. And they've taken everything in the German Azure datacenters, and they've put it in the hands of a different company, Deutsche Telekom. And the way that the agreement is written is that Microsoft doesn't have legal or technological access to any of the data. And the Azure center itself, their setup keeps any of the backups, any of the replications, the high availability or the fault-tolerant copies from every going to another Azure cell in another country. So everything remains under German law, under German jurisdiction. And Microsoft is doing this rather than coming up with unbreakable encryption that they could then go to the U.S. government and say we can't break it, and then the U.S. government coming back and saying, well, you're going to help us. They can now say we have no legal recourse to get the data that you want. It's not in our hands. It's not in our datacenter. You have to go talk to Deutsche Telekom, which means you have [crosstalk].

Steve: Ah, so they put it out of reach.

PADRE: Exactly, put it out of legal reach. And Microsoft is saying, well, if they can't understand the math, they do understand legal jurisdiction.

Steve: Yeah.

PADRE: You know what, Steve, this story, I know a lot of people have had enough of it. So let's stop talking about Apple. Let's stop talking about Apple products. What's the next story?

Steve: Okay. So I'll just say that we do keep trying to stop talking about this, but this is the story that just keeps on giving.

PADRE: Yes, it does.

Steve: I mean, it's just one thing after the next. And there's no way we're not going to be able to talk about the Burr-Feinstein as soon as we see that because, you know, a bill is not a law. A bill is a wish. And then it's got to go through both houses of Congress, and then the President has to sign it into law. And although Obama has been sort of seeming a little more willing to do that, he made that comment at the SXSW event where he talked about people fetishizing their phones or the security of their phones. And it's like, whoa, okay.

PADRE: Yeah.

Steve: Anyway. So we do need to talk about actually still Apple with something that happened with iMessage. Again, Matthew Green, the Johns Hopkins well-known cryptographer, he admits in his blog posting about this that he's got a thing for iMessage. It intrigues him. It interests him. Part of it is that it is such a heavily used protocol. Part is that it has been reverse-engineered by a third-party entity. And the point is there's enough known about it that he's been able to poke at it.

So yesterday Matthew Green wrote the following, and I'm just going to read the first four paragraphs of this because it also really, you know, this is coming from one of our leading cryptographers, and it helps to sort of frame the way he feels about this in general. He wrote: "Today's Washington Post has a story entitled 'Johns Hopkins researchers [meaning him and four students] poke a hole in Apple's encryption,' which describes the results of some research my students and I," writes Matthew, "have been working on over the past few months. As you might have guessed from the headline, the work concerns Apple, and specifically Apple's iMessage text messaging protocol. Over the

past months my students" - and he names the four of them - "and I have been looking closely at the encryption used by iMessage in order to determine how the system fares against sophisticated attackers. The results of this analysis include some very neat new attacks that allow us to, under very specific circumstances, decrypt the contents of iMessage attachments, such as photos and videos."

He says: "Now, before I go further, it's worth noting that the security of a text messaging protocol may not seem like the most important problem in computer security. And under normal circumstances I might agree with you. But today the circumstances are anything but normal. Encryption systems like iMessage are at the center of a critical national debate over the role of technology companies in assisting law enforcement. A particularly unfortunate aspect of this controversy has been the repeated call for U.S. technology companies to add backdoors to endtoend encryption systems such as iMessage.

"I've always felt," writes Matthew, "that one of the most compelling arguments against this approach, an argument I've made along with other colleagues, is that we just don't know how to construct such backdoors securely. But lately I've come to believe that this position doesn't go far enough in the sense that it is woefully optimistic. The fact of the matter is that," he says, "forget backdoors. We barely know how to make encryption work at all. If anything, this work makes me much gloomier about the subject."

And he says: "But enough with the generalities. Apple iMessage, as implemented in versions of iOS prior to 9.3," which is yesterday when we got 9.3, which not coincidentally coincides with the public revelation of this. So he was waiting until 9.3 was released because Apple made some changes which mitigate the problem. But as we're about to discover, there is a fundamental flaw in the design of iMessage which at first looked safe and proper, but is exploitable and is not fixable without scrapping iMessage. And in fact he recommends toward the end of his blog that Apple really ought to abandon the iMessage protocol and switch to something that is known and has been analyzed publicly like the technology used by Whisper Systems' Signal application. And so this is interesting because Apple designed their own protocol. And even with really good crypto people, they made a mistake, which is what Matthew and his team found.

So he says, prior to iOS v9.3 and Mac OS X v10.11.4, the previous versions contain "serious flaws in the encryption mechanism that could allow an attacker who obtains iMessage ciphertexts" - meaning encrypted iMessage communications, which is what's passing back and forth between devices - "to decrypt the payload of certain attachment messages through a slow but remote and silent attack, provided that one sender or recipient device remains online. While," he writes, "capturing encrypted messages is difficult in practice on recent iOS devices, thanks to certificate pinning, it could still be conducted by a nation-state attacker or a hacker with access to Apple's servers." He says: "You should probably patch now." Meaning update your version of iOS on all of your iOS devices and your Mac OS X.

So essentially, and I'm going to summarize this, we've discussed on the podcast how in order to protect the secrecy and authenticity of encrypted messages, it's necessary to both encrypt and authenticate the contents. And we've talked about how, since you're doing two things, if you're doing them separately, that is, you often have an encryption algorithm and then a keyed MAC, often an HMAC - and MAC is Message Authentication Code. So the idea is that you take the plaintext. You encrypt it first, and then you authenticate the encrypted result and do that after encryption. And the reason the sequence is important, is then the recipient, the first thing they do is to authenticate that there's been no change to what was received. And this is something that people often get wrong because they don't believe it matters what the order is.

But as we've discussed previously, if you authenticate first, then you encrypt, what that allows an attacker who is able to receive that payload to do, they can perform test decryptions because the outer layer is the encryption layer, and the inner layer is what was authenticated. And so what Apple did in their design of iMessage is, unfortunately, they don't authenticate. They thought they were because they were digitally signing the result. And signing is, as we know, is typically a hashing process where you take the blob, you hash it to produce a signature, and then you cryptographically sign that signature.

Well, it turns out that, looking carefully at the protocol, Matthew and his students realized that they could take the payload and change the signer, essentially, the attacker could sign it, and that there was a way to then abuse the endpoints. This is why one or the other, either the sender or the recipient, has to stay online to essentially send test packages back with a valid signature, but a different signature. And since they control the signing, they can make the signature valid. And over time this allows them to essentially use either those endpoints as what's called an "oracle," which is essentially - that's a cryptographic term for sort of getting information from, as it's called, an "oracle" by sending test packets and looking at the type of error generated.

And so this is the kind of attack we've talked about for a number of years. And it turns out that because the design of iMessage wasn't, I mean, it looked good, and it's easy to examine it and say, yeah, this looks fine. But because it broke from the well-agreed-upon proper way to do this, just that little difference created a crack that it turns out you can drive a wedge into. And the bad news is that the only fix that anyone was able to come up with, because Matthew and his team have been in contact with Apple for some time, I mean, this is a fundamentally compromised protocol. And it's in all billion iOS devices right now; and Macs, of course, have it, too. So it's not easy to change them all.

The only thing they could do was essentially create a cache and use a cache of traffic to detect attacks, which is just like the most horrible kludge imaginable. The only thing they could think of to fix this, I mean, to call it a mitigation, is essentially an intrusion detection system for iMessage that would work by caching traffic and detecting this kind of exploit, which is not the way you want your secure protocol to function. It's like, well, yeah, we know it has problems, and we can't change it because it's in too many devices now. But we know how to detect it if we store up enough activity and check to see if this is repeated probes by a given attacker against a given device.

PADRE: And what they'd be looking for in the cache, they'd be looking for the same packet that has been sent time and time again with a slightly different signature. Let me layperson this because I think I understand it, but let me make sure. Because Apple decided to authenticate after encryption, it allows an attacker to change the contents of the packet and to send it back at either the recipient or the sender and listen for error messages, basically looking for a change that they make that will finally get back what they want. Now, if Apple had done it the other way, if they had authenticated, then encrypted, then there would be no way for an attacker to change the packet without breaking the encryption and making the packet useless.

Steve: Well, except that authenticating, then encrypting, is the other problem, which allows you to then do test decryptions. Essentially the mistake they made was they authenticated by signing. And so normally authentication uses a fixed authentication key which is agreed upon. Instead, they authenticate under a signature. So what this allows is them to change the signature. They use the attacker's signature to sign it, so the signature is valid. But that allows them to then poke at the authentication side and, over time, end up decrypting. And there is a paper that's been written, if anyone's interested, that really chews this thing to pieces. There's no need to go into it here.

Essentially the takeaway is there is a problem. It's not clear what Apple will do in the long term. What would be nice is if they could bring up a second publicly vetted messaging system like Whisper Systems' Signal, and sort of phase it in so that it's able to be adopted by devices that use it, and devices that haven't upgraded are still able to use iMessage at their end. But over time, iMessage would filter out of the ecosystem. Unfortunately, that sounds very much like SSLv2, which is the topic of this podcast as soon as we get caught up on the news, and we see how well that worked. So it's a problem that, I mean, there is a problem with iMessage, and it can't be changed.

PADRE: Yeah. Unfortunately, what it looks like is iMessage is fundamentally broken.

Steve: It is.

PADRE: And as you mentioned, because it's in so many of their products, you can't fix it without breaking it for a large number of existing iOS or OS X devices. So the only way to fix this is either to use that kludge, which has all sorts of negative consequences, as well, because you are now making a cache of data that should never be cached; or you junk it, junk it and start new.

Steve: Well, yeah. And, I mean, it just - anyone hearing this would just shudder because you don't - you want your encryption to be strong. And unfortunately, it's not. And so they're sort of like coming up with a pure kludge in order to detect the attack, rather than foreclose the attack, to prevent the attack. Maybe they'll be able to run parallel protocols and get this out of the system over time. And in his blog Matthew notes the phenomenal number of iMessages. Apple boasts a billion devices. And I think it was something like 200,000 iMessages per second, something like that. I mean, it's an unbelievable amount of messaging traffic that Apple is now handling, none of which is as secure as they would like it to be.

PADRE: Steve, any guesses on how many messages you'd have to have in the cache in order to do proper IDS/IPS?

Steve: I don't know. I didn't look at the paper to see whether they talk about mitigation. The problem is you'd have to have a lot because the cache is going to be flushed. And if it's got that much traffic, any of that traffic could be an attack hiding among the rest. Oh, it's [groaning].

PADRE: And of course, once the attacker knows that you're looking for a single packet that's being used many, many times, all they have to do is, okay, now they take a selection of packets, and they alternate which ones they send.

Steve: Right.

PADRE: So it becomes a cat-and-mouse game with a kludge.

Steve: And maybe they slow the attack down so that they're - if they know that the mitigation is a cache, they wait for the cache to roll off the end so that their attack ends up being a low-bandwidth attack in circumstances where they know the victim device will be online for, like, you know, statically online and not in danger of going away. Then, I mean, it just - these are all the reasons why this is just an awful solution.

PADRE: Yeah, yeah.

Steve: Ow. So we've talked often about side-channel attacks. And in fact a couple weeks ago we covered one that I mentioned where a laptop on the other side of a wall had its, I think it was GPG keys stolen when it was doing - it was a chosen ciphertext, meaning that it was necessary for that laptop to repeatedly decrypt a message of the attacker's choice. That gave them the behavior that they were able to literally pull out of the noise that was received in order to determine what was going on.

And the reason was that, even though the specific crypto had been designed such that no branches were taken based on secrets, that is, the keying material didn't cause execution path changes, it did cause memory usage changes. And they were actually able to detect the effect of timing on cache hits and misses by running a thread in the same core as the one doing the decryption. And I just realized I'm confusing two different side-channel attacks. That was one where there was a thread in a hyperthreaded core. And then the one we talked about earlier was the through-the-wall attack.

This one was designed, this most recent one, by some researchers at the University of Tel Aviv. And essentially what they showed was that on mobile devices with very inexpensive hardware - and this is really the advancement in the art in this case. Doesn't take fancy software-defined radios with multiple antennas, and you need to get all of the conditions set up exactly right. This is a hand-wound coil that's connected to a soundcard because it turns out that it only needs a bandwidth of a few hundred kilohertz, which the A-to-D converter in a soundcard can handle. And that gives them the resolution that they need in order to perform this decryption.

In the abstract of the paper which they have submitted, they wrote: "We show that elliptic curve cryptography implementations on mobile devices are vulnerable to electromagnetic and power side-channel attacks. We demonstrate full extraction of elliptic curve DSA secret signing keys from OpenSSL and CoreBitcoin" - by the way, CoreBitcoin is the library which a bunch of different Bitcoin clients use - "running on iOS devices, and partial key leakage from OpenSSL running on Android and from iOS's Common Crypto library." They write: "These nonintrusive attacks use a simple magnetic probe" - which is what they call their coiled wire - "placed in proximity to the device, or a power probe on the phone's USB cable. They use a bandwidth of merely a few hundred kilohertz and can be performed" - "they" meaning the attacks - require "a bandwidth of merely a few hundred kilohertz and can be performed inexpensively using an audio card and an improvised magnetic probe."

So here, I mean, and you kind of wonder, it's like, wait a minute, maybe that affects a phone which is on and locked. And that seems like it would be useful for law enforcement. And the paper just came out.

PADRE: And we know that a side attack, a side-channel attack is attacking crypto by not attacking crypto, but attacking the physical implementation of the crypto, and that's why we've got that coil. So what is it that they're listening to specifically?

Steve: So essentially we all know that our electronic devices emit electromagnetic radiation, which is why, if you look inside modern computers, you'll see sometimes like a sponge-y silver gasketing around the box, literally creating a Faraday cage in order to keep the electromagnetic radiation inside. And oftentimes the power supplies on laptops will have a big blob in the power cord near where it plugs into the computer. The idea is that that would be a means for radiation getting out of the computer through its DC power cable so that large ferrite toroid, essentially, around the cord absorbs that interference and prevents the long wire coming from the laptop from being an antenna to re-radiate this radiation.

So the problem is that it's extremely difficult to prevent any consequence of the secret activities of the electronics from leaking what they're doing. What they're doing creates radiation. And clever engineers have figured out, first of all, how to pick up the radiation, and then separate essentially the wheat from the chaff, separate little bits of information, literally bits of information out of the noise when the processor is actively decrypting or processing encrypted information. It leaks.

And this kind of thing, these sorts of side-channel attacks go way back. What was it, it was the Tempest project where it was realized that the beam scanning a CRT, a cathode ray tube, the beam was changing its strength in order to paint the image on the CRT. And in doing that, it was emitting electromagnetic signals, and somebody could pick that up from a goodly distance and essentially re-rasterize it in order to recover the image that is being shown on the CRT. And so the fact that I'm using the term "CRT" demonstrates how old these sorts of attacks are. And we have them today, using a coil of wire laying against a phone.

So what this potentially means is that a phone engaged in cryptographic activities, sitting there with its screen off and locked - because phones still need to be able to transact data, to send and receive email and operate in the background, many phones do - a coil of wire next to the phone can pick up data and figure out what's going on inside.

PADRE: You know, Steve, when I was in high school I did an internship. And it wasn't any top secret stuff, it was corporate secret stuff. But we had a room that was entirely encased in copper mesh.

Steve: Wait, was that high school?

PADRE: Yup.

Steve: You had a Faraday cage when you were in high school?

PADRE: Not at the high school. It was a company that I was working with in the Silicon Valley. No phones. The only line that came in was power, and it was filtered power. And every device that was in the room had to stay in the room, and you actually came in in clean suits. And so that was [intentionally muffled] years ago. And so even back then they understood, yeah, you can piggyback signals, and if you want to keep something really, really secret, you need this for the systems that will be containing your most sensitive secrets.

Steve: Right. And in the movies we see situations where somebody has a big bin, and they say, okay, everybody put your phones in here, and those are not coming in with you. You're coming in without your devices because there's no way to secure them.

PADRE: Actually, the last movie I remember that actually had a Faraday cage in it was - you remember a movie with Will Smith and Gene Hackman called "Enemy of the State"?

Steve: Yeah.

PADRE: And Gene Hackman had like a tractor-trailer that had copper mesh on the inside. And I was going, okay, well. And I actually priced out one of those. Those are actually very expensive to make.

Steve: Well, and in fact there's a series now on Showtime, "Billions," which is really good. And just last Sunday, and maybe the Sunday before, in order to have private

conversations, knowing that they were probably being bugged and eavesdropped on, they walked into an electromagnetically shielded room in order to have that conversation. So just two days ago this was happening, this whole notion of a shielded room to prevent what goes on inside from getting out. Of course, it doesn't prevent recording from getting out, so you still have that problem. Only electromagnetic radiation.

PADRE: We built one at Interop one year just using stripped copper wire from old CAT5 cables. A, it was very, very difficult to make properly because you also have to ground the thing properly in order for the signals not to just get attenuated and then continue on their way.

Steve: And the size of the holes determines what wavelength it's going start attenuating. And so it really needs to be - what you want is sheets of copper.

PADRE: Yeah, yeah. And actually that was very expensive back then. So we never completed it. Go figure.

Steve: Yeah. So...

PADRE: Cleansing breath. Cleansing breath. Okay. So our messages are being spied on. Our phones are being spied on. Our computers can be attacked from a distance with a coil and a cheap soundcard. You've got some good news; right, Steve?

Steve: Not yet, no.

PADRE: Okay.

Steve: It also turns out that the ad networks used by a bunch of major websites have been spreading everyone's biggest nightmare, crypto ransomware. The SpiderLabs blog discovered through their watching the 'Net that the Angler - Angler is sort of the go-to evil toolkit now which is getting into systems and spreading crypto ransomware. At the beginning of this month a malvertising campaign appeared; and, based on their research, it exposed tens of thousands of visitors, even just in the course of a day: The New York Times, the BBC, MSN, AOL, Xfinity.com, NFL.com, Realtor.com, TheWeatherNetwork.com, TheHill.com, Newsweek.com, Answers.com, ZeroHedge, and InfoLinks. So, I mean, a Who's Who of major sites. And the affected advertising networks were those owned by Google, AppNexis, AOL, and Rubicon.

When looking at the payload, it turns out that there was a JSON-based file that was being served in these malicious ads, containing more than 12,000 lines of heavily obfuscated code. And when this code was decrypted and deciphered and deobfuscated, researchers discovered that it enumerated a long list of security products and tools which it was actively avoiding in an attempt to remain undetected. So this was not just a malvertising campaign, but it was trying to be as stealthy as it could be in order to allow it to survive as long as it could.

And what I thought was interesting was that the most widely seen infection domain name was a domain called BrentsMedia.com. And the WHOIS record showed that it had been owned by an online marketer until the first of this year, till January 1st, when that domain was allowed to expire. It was snapped up by its current owner on March 6th, which was a day before the malicious ad onslaught started. So that was sort of interesting. Here was an existing domain name. It had been retired. But it had some credibility. It had some rep. And so the bad guys got it in order to leverage its history,

essentially, and use that for spreading their malware.

And so in the researchers' report about this, they asked themselves, what can people do? And the standard advice is what we've been giving all along, is you want to reduce your attack surface by uninstalling things like Adobe Flash and Java and Microsoft's Silverlight, and really any other third-party browser extensions that you're not actively using. One of the things that can happen is that we just, you know, these things just sort of get acquired over time. It's like, oh, this looks interesting, and so you add it, but you don't normally go through and do a proactive housekeeping to say, you know, am I really using Pocket anymore? Am I really using this or that?

And it's just better to keep things simple. It's better not to have add-ons around because, if vulnerabilities are discovered in them in the future, and you've got it, and you're not using it, then you're presenting a larger attack surface than you need to because you're not getting any benefit from that attack surface. So you really want to minimize it as much as possible. And of course the other advice is this tension that now exists between sites that are becoming increasingly insistent that their visitors disable any adblocking, and the surfers, the visitors to those sites who don't want to be infected with malware in return for visiting.

So by default - and this is why we've been recommending now uBlock Origin instead of NoScript. For years, NoScript was what I was promoting, just turning scripts off so that this stuff wouldn't be allowed to run. The problem is good scripts became so prevalent over the last couple years that too many sites are broken; or the site works, but then you go to fill out a form; or you're in the middle of moving through the purchasing system, and then it breaks, so now you have to resubmit a form that you really don't want to resubmit because you don't have scripting.

So finally I just gave up on NoScript and blocking scripting and went to really the main source of this. Although uBlock Origin is also a script blocker, it focuses on, by default, allowing you to set up a set of blocks to protect you. And then, if you need to, you lower your shields, if you decide that, okay, this site is asking me to, or I want to support the site, or the site has put up its own ad wall saying you can't come here until you allow us to serve ads to you.

PADRE: It's interesting that the whole adblocking thing, the controversy that erupted towards the end of last year was really about whether or not you're going to support the sites that you enjoy. So if you support the site, you should disable your adblocker. But it's really morphed into, well, since you don't maintain the quality of the ads that are served on your site, I can't trust you. And it will be interesting to see if newspapers and online publications will start moving to a format in which they have to vet a company coming into their ad network before they actually allow them to display ads on their site because it's their reputation.

Steve: Well, The New York Times and the BBC sites were hit. So high-end, high-profile sites. And I've never really looked closely at the ecosystem that the ads use. But the way I understand it, there's actually some sort of a bidding process where advertisers have a certain number of impressions that they will pay a certain amount for. And so they're not even sure where they're going to be shown, or how much they're going to be shown. It's very much like even here on the TWiT network we'll have an advertiser at the beginning of the year that ends up using up its number of allotted sponsor inserts, and then we switch to different sponsors. And so it is a dynamic process.

And so I think, of course, the happy world that websites have existed in until now is that they just created blank space. They just had windows on their pages with, we don't know

what's going to be put in there, and we don't care because we're getting money by creating a sort of a window of opportunity for ads to appear. And sadly this has become a major target for abuse. What we really need, and a project I'm going to have to find some time to spend, is we really need stronger browser isolation. We need a browser that - it is the vulnerable attack surface in our computers today. That needs to be in a virtual machine where nothing running in there has access to the crown jewels, outside of that virtual machine.

And there are tools like Sandboxie. There's something called Browser in the Box. There are different approaches at trying to do this. And there are some services. I'm not a big fan of the pay-by-month service approach. I think a solution needs to be either free or something that you just pay a reasonable amount for once and then it's yours. But we really - we need a good solution for that. And you need to, when you want to export a file that you've downloaded from that containment, then you need to be able to do that. So it can't live by itself because it does need to have - you need some transaction between the browser and your system, but under your control, not under the control of scripts running in ad windows. And if we had that, then we could drop - except for the privacy and tracking side. That's still an issue. But at least the malvertising would cease to be a problem.

PADRE: The way that I have it running at home is I actually have a box sitting in my data closet that all it does is run containers. And every time I bring up a browser on my laptop or desktops that are inside my network, it's not actually bringing up a browser on the laptop or desktop. It's bringing up a container that is shown on my laptop or my desktop. I can get that infected 50 ways to Sunday, and when it shuts down, it disappears.

Steve: Nice.

PADRE: And because it's running in that container, it doesn't know about anything else that is on that desktop. In fact, it doesn't even know anything else about the computer it's actually running on. You can do that on a single system. You can run containers in your operating system, be it Windows, Linux, or OS X.

Steve: Right.

PADRE: So I think that's where we're heading. We're heading to, as you mentioned, something that they covered at RSA two weeks ago, which is assume breach. And if you assume breach, it changes the way that you use your system.

Steve: Yes. Well, and speaking of the Angler exploit kit, which is the thing that all of this malvertising was dropping onto machines, onto visitors' machines, we've been talking off and on about the ongoing problems that Lenovo has been having with all, you know, like installing certificates in people's machines and really creepy behavior. The ThinkPad line was my favorite go-to laptop for years. And of course that was when IBM was producing those. They were just - you've seen executives with a ThinkPad. It's like, okay, this is a solid laptop. And then of course...

PADRE: Quality hardware. It was relatively clean on the OS side. And then it just - they threw that all away.

Steve: Yes. So it turns out that F-Secure detected on Startpage.Lenovo.com an instance of Angler that visitors to Startpage.Lenovo.com were being exposed to. So certainly this was not deliberate. Their problem, Lenovo's problems, were of their own making. Presumably this is a site compromise. Somebody got into their site somehow and altered

that page in order to send Lenovo's visitors off to go pick up a copy of Angler and get themselves infected with malware. So still bad.

PADRE: It's one of these things where I actually kind of feel bad for them because they went from a premium brand to the brand to avoid at all costs in about 18 months. And I know several of the engineers over at Lenovo, and they're good guys, and they put a lot of work into this. And then it seems that it gets to the marketing folks, and the marketing folks say, well, let's tack on an extra three cents of profit per unit, and then this happens.

Steve: Yes. And in fact some of this was third-party crapware that they were being paid to bundle with the laptop. And you see, you know, Symantec preinstalled or McAfee preinstalled, and you try to remove it. And it's like, wait a minute, you have a license. It's like, yeah, for a month, and then you want me to install it or start paying for it.

PADRE: That's my favorite warning. Every time I get a new machine, and I go into the uninstall menu, and it's just boom, boom, boom, boom, at least 15 different times I get the "Are you sure? I mean, you have an active subscription. Don't you want to take advantage of the industry standard in security?" It's like, no, thank you, go away.

Steve: Yeah. And I also don't want - what was it the other day? Something was pushing Chrome on me. I don't remember what it was. And it's like, yeah, I'm still sticking with Firefox.

PADRE: I haven't seen that one before. A site was pushing Chrome?

Steve: No, no, no. It was - I don't think it was Java. I don't remember what Java's pushing. Something was.

PADRE: I'll have to try to replicate that now.

Steve: Anyway, Pwn2Own 2016. This is the annual everybody saves up their exploits and gets together and attacks operating systems and browsers contest. This year about 100,000 less dollars were earned. In the 2015 Pwn2Own, $550,000 in cash prizes were awarded. This year only, but still, 460,000, so some serious money, as a result of 21 new vulnerabilities which were discovered and revealed and demonstrated. Windows 10 was the leader in vulnerabilities with six of them. It was the most in any single target of the competition. However, in fairness, OS X, Apple's, was not far behind with five. So both of these big operating systems.

And, I mean, we don't want our operating systems to have vulnerabilities. But you look at something like Windows 10, and my first thought is only six? How hard did these guys look? I mean, because these operating systems are just ludicrously large and complex now with just, you know, I mean, anything that's going to squat on 20GB of your hard drive and is full of stuff it's doing, it's going to have vulnerabilities. We see how even simple clear protocols are having vulnerabilities found. So something like Windows 10, yeah, of course it's going to have problems. Anyway, Safari had three successful attacks against it.

And what was interesting to the people observing this, sort of standing back and observing, is that Microsoft's Edge was significantly more secure than previous versions of IE. So the decision Microsoft famously made, as we know, was to abandon the aging IE codebase and just say we're going to start over. And so that looks like they made the right decision. Edge is way less problematical than IE was. And Chrome remains the most

secure of all browsers. The observers noted that one of the reasons Chrome is as strong as it is, is that it enforces really good sandboxing at the expense of memory footprint.

And, for example, that's why I still shy away from Chrome. When I launch Chrome, because I'm watching, I've got TaskMan running on a monitor all the time, and I just see chunk, you know, memory utilization jumps up by about a third of my system's memory, which is one of the reasons I will be moving to a 64-bit system soon. I'm still on Windows XP, where I've only got 3GB of usable memory. So, yeah, Chrome takes a big chunk of that. But, I mean, it is just really memory hungry. But what you get in return, arguably, is the best security in the industry.

And what was interesting was that Firefox was not present at all. That is, people didn't even bother attacking it. And observers felt that this was because, among elite hackers, Firefox is no longer considered hard enough to beat, like no one gets any great props for breaking into Firefox. And the problem of course is, as was the case with IE - and remember, this is Mozilla. This predates Internet Explorer in terms of it was the first browser on the Internet, and no doubt there's no remaining code from then. But still, it's arguably time for a rewrite, just as it was with IE.

Mozilla does have their memory-safe language Rust. And people are beginning to feel that, because Firefox doesn't have really strong sandboxing the way Chrome does, and it's got an aging add-on architecture, whereas Chrome has a much better extensions architecture, that it's probably time to say, okay, we're just going to come up with a rewrite and get back in the game. I love Firefox because I've got tree-style tabs that allow me to have tabs running down, a scrolling window of tabs running down one side, so it fits the whole wider screen structure that PCs and laptops have now, that is, to take tabs, to have tabs on the side, and to have lots of them.

And of course there's a great ecosystem of goodies that I like to add to Firefox; although, in fairness, Chrome has those, too. I just wish we could get good side tabs with Chrome. I've seen something that sort of puts another external window next to Chrome. But it doesn't seem like it's the same as actually having just side tabs built in.

PADRE: Well, I mean, that's why I have a desktop with 128GB of memory, because I like to use Chrome. People ask if it's for gaming or video editing. No, it's just for all the tabs. Steve, on Pwn2Own, what's interesting is the big bounty this year, and I don't think it was claimed, was for escaping a virtual machine and gaining root-level remote code execution. And that was not claimed.

Steve: Good.

PADRE: So evidently virtual machines are still pretty secure.

Steve: Now, we have seen flaws.

PADRE: Yes.

Steve: The one that comes to mind was the floppy driver flaw where, even though - and this is one of those things, again, attack surface. You are not using the floppy driver in a virtual machine, probably. Yet it's there. And it turns out that there was a way to exploit the floppy driver in order to break out of the VM encapsulation and get out into the host environment. And of course we have seen cross-VM exploits where multiple VMs could communicate by somehow breaching the hypervisor. But it's good to know that that's strong. I mean, it just - it feels like the right solution.

This notion, for example, that Sandboxie has of intercepting APIs that are dangerous, it's like, okay, that's like old-school firewalls where you block the ports that have had attacks come through. And it's like, okay, yeah, but what about those other 63,000 ports? When is an attack going to come through one of those? The idea of blocking things that are dangerous is not the way you want to operate. You want to block everything by default, and then selectively and under deliberate control pull things out of that container.

PADRE: Exactly. And that's the way we should be operating, but we don't because that requires more work, and that requires an active decision to allow something, whereas we would prefer not to deal with it unless we think it's causing a problem.

Steve: Right.

PADRE: That's human nature.

Steve: Right. So we've talked about Stagefright too much in the past because it just keeps giving. And that's certainly the case here. What's happened is Zimperium, everybody will remember, were the guys who found a number of flaws in the media processing code of the Stagefright module, which handles multimedia for Android. Google didn't write it. They got it from somewhere. And it just turns out it has all kinds of problems. And so we've talked about those. Zimperium created a tool that lists all of the known vulnerabilities. And over time then that brought out the whole issue of are mobile phone manufacturers maintaining phones after they've sold them? Do they care about them? The argument being, you know, these are computers. We know computers have to be kept patch current.

And so on one hand the mobile suppliers want to just consider it a retail sale and then charge you for connectivity. My argument is, if they're profiting on the service of you being connected to them, then with that should come an obligation that this device they sold you and are using to connect and consume the service that you're paying for, they ought to have responsibility for pushing to it patches which are being made available to them. It's not like they've got to independently create a security team and design fixes. No, all they have to do is push what security researchers or often Google makes available. They're lazy. They choose not to. Or the other problem is they may say, oh, well, we'll do it for a year. Well, they ought to do it for the service life of the device. If they're taking revenue from you for using the device, they should have an obligation.

Anyway, what's happened is a different group, an Israeli security firm named NorthBit has come up with a new exploit against the old Stagefright, that is, the patched Stagefright, which uses a vulnerability which, as it was written, for some reason Google never fixed. Apparently this was a known vulnerability that never got patched. So what's interesting is that, in Android 4.1, Google increased the security by adding address space layout randomization {ASLR}, which we've talked about often. This is a technology that Windows got a while ago, and OS X, or even before OS X, Apple's Mac OS got, where the system is inherently composed of a bunch of individual modules.

And the dangerous way of arranging things is just to stack them all from the bottom of memory up in the same order every time. That's dangerous because what that does is it creates a rich codebase of repurposeable code which attackers can use. If in applications running elsewhere in memory, they can get any kind of a foothold. If they can create a buffer overrun; if they can leave something on the stack which gets popped which causes the return, not to the function which called the malicious function, but a return to existing code at a known address elsewhere in the system. And that's called ROP, Return Oriented Programming, where you're able to stitch together existing code to perform

some malicious feat.

The reason that's important is that those regions of memory are marked executable, whereas the stack is typically flagged as being data only. It's called the NX, the No-eXecute bit. That can be set on the descriptor of the memory for the stack, which prevents the attacker from running their own code that they may have brought into the buffer, but they can still cause a fault where existing code that's authorized to run in the operating system can go. So in order to defeat this, the concept has been, hey, there's no reason we need to stack everything from the ground up in the same order each time. Let's just scatter all these modules for this modular operating system. Like in the case of Windows DLLs, these Dynamic Link Libraries, they're dynamically linked. They can be dynamic anywhere they want to be. So let's just scatter them around.

Now, there are normally limitations on the granularity of position, which brings the entropy of their location down some. But at least the idea is that, every time the system boots, it arranges somehow to randomly or pseudorandomly rearrange the modules so that code that is then running is not at a known location. And we've seen exploits where the bad guys just, okay, we're in an ASLR system, so there's a 1/256 chance that the module we need to execute code from will be in the location we're hoping. If not, we crash. If so, we win. And so the attackers basically accept a much lower degree of success on their exploits, specifically because of ASLR.

Well, it turns out that this, it's called the "Metaphor" exploit, is able to leverage an information leakage due to the way this Stagefright module, the flaw in the Stagefright module, to obtain the exact layout of modules in the ASLR system, which completely removes the effectiveness of address space layout randomization and hugely improves their ability to successfully attack the phones. So there's a proof-of-concept exploit which works against Android versions 2.2 through 4.0, doesn't work against 4.1, but does against 5.0 and 5.1. And in fact it was believed that 5.1 was immune, but it's believed to be that no longer. It turns out that 19% of Android phones are currently running v5.1.

Now, it does require phone-specific code. So one so-called drive-by exploit won't be able to attack all Android phones, even of the same version. But it turns out that a site, for example, could see what you've got and then send you a phone-specific payload to attack that particular phone. And the researchers said that this works best on Nexus 5 models with a stock ROM; also on the HTC One, the LG G3, and the Samsung S5.

So Stagefright is back. And I saw a number somewhere, I had it in my notes. Oh, nearly 300 million Android phones. The original Stagefright exploits were effective against a much larger number, 950 million. So this is less than a third, yet it's a much stronger attack. So if you had the at-risk phone with the not-yet-patched version of Android, and the payload was for that phone, it looks like the attack would be pretty successful.

And then just in late-breaking news, this was just yesterday, Google has stated in response to this, they wrote: "Android devices with a security patch level of October 1st, 2015 or greater are protected because of a fix released for this issue last year." And then Google said: "As always, we appreciate the security community's research efforts as they help further secure the Android ecosystem for everyone." So our moral is keep mobile devices updated. Only use mobile devices from reputable vendors who do keep their devices updated post-sale, and use them only so long as they are being kept updated because these things are computers, and all the evidence demonstrates that they're going to have problems.

PADRE: Steve, the ASLR leak, so it's leaking where the target module is in memory? Is that the bit of information that's leaking out?

Steve: Yes. It's described as an information leakage. And I forgot to mention I've got the link to the exploit description in the show notes. I was stunned. It is an absolutely amazing piece of work. If anyone is interested in a case study in how to do this, how this works, these guys, these NorthBit guys, it's, well, the domain is Exploit-db.com/docs/39527.pdf. It's beautiful. It's got graphics. It walks you through this thing. It's just a beautiful piece of work. So if anyone is interested in, like, digging in, these guys have really nailed it. But to answer your question, Padre, yes. Basically there is an information leakage which discloses the module location instance, which then allows the bad guys to attack with essentially 100% reliability. Basically, if everything is set up right, you get owned.

PADRE: I just started looking through this file. It's interesting because it looks like they're exploiting the - they've got an unsigned integer, a 32-bit unsigned integer that they declare, and then they put it into an unsigned 64-bit in order to give it extra space. So all you have to do is write 65 bits of data, and you've got an overflow.

Steve: Yup. It turns out it's in the MPEG-4 file format implementation. There's just a mistake there in parsing it. And we saw that was Stagefright before. There were some typecasting problems where the compiler just didn't do - it wasn't checking for inequality correctly. And so it was like signed versus unsigned. And that difference in inequality testing allowed them to, like, sneak past a test that was there to keep this from happening. It's like, well, the test is there, and we just bypassed the test.

PADRE: What I do like about this is in the structure that they have in the example here, it basically tells you how to overflow the module because it has what the input is, and then it has a separate line that says, oh, yeah, 64-bit, it will never get bigger than this. It's like, oh, okay. I'll make it bigger than that.

Steve: Right.

PADRE: Thank you for that very helpful comment.

Steve: We're not going to bother checking because no MPEG-4 file is ever going to have something bigger than 64 bits.

PADRE: A 32-bit unsigned integer will never be bigger than a 64-bit unsigned integer; right? I mean, let's just...

Steve: And think of all the time we'll save not performing a test.

PADRE: Now, that's performance. There we go.

Steve: Wow.

PADRE: Wow, okay. Thank you for showing that to me because now I have my evening reading.

Steve: You'll dig it. It's a beautiful piece of work. So last week I mentioned that the way I had set up my new monster box - and Padre, I went with 64GB because I wanted to crank the memory up to 3200. I can run it 2666 at 128GB of RAM, but I decided, you know, most of the time the RAM is going to be empty. So I think I'd rather have what I'm using more. I do have the unused memory, in case I ever am doing something like going VM crazy. But anyway, what I did was I did get a motherboard that has the M.2 PCIe slot and the state-of-the-art Samsung 950 NVMe memory.

PADRE: NVMe, yes, ooh.

Steve: So half a terabyte of that. So it's 512GB of that, essentially on the motherboard. Nothing faster. And I still operate lean. That is, I'm right now in XP. I'm in a 30GB partition. My C: drive is like, you know, maybe half full. So my idea was to bite the bullet and get a much larger SSD than I know I will ever use. That is, only use a corner of it. That way wear leveling helps me because I'm not filling this thing up. Instead, as I use my small amount of it, the controller will migrate me around, will migrate that small usage around a mostly unused field of nonvolatile RAM. So I'm minimizing my use of it. But then the other part of this I mentioned is that I am using a motherboard that supports the Intel RAID 0, 1, and 5. I think also 10 and a couple others. I'm using just a simple mirror.

I wasn't sure I mentioned - I thought I had two 2TB drives. Actually it's two 3TB drives. I had them as backups for my TiVo in case the 3TB drive in the TiVo ever died, I'd have one on-hand that I could immediately swap in to get that going again. So what I've done is I have those mirrored so it's maximum redundancy, minimal efficiency, but also maximum speed because any, for example, RAID 5 or 6, there is some performance penalty hit that you take unless you have a fully standalone caching controller because of the math you have to do in order to perform the XORing operations to create a more efficient form of redundancy that the higher levels of RAID give you. I'm settling, I mean, 3TB is a crazy amount of space, way more than I expect to need. Again, I'm operating just in hundreds of gigs right now across the drives that I have online.

So what I'm going to do is, every night, because I never turn my systems off, my main system just sits here running 24/7, I do use my own little Wizmo in order to power off all of the screens. I've got five screens running. And this one has two quad HDMI cards, fanless, so that I've got eight, I can drive eight screens out of those. It's just an NVIDIA card that's, like, all heat sink but no fan. So I'll have plenty of screenage for the foreseeable future.

But the point is that I - and I mentioned this on the podcast - that every night a full image of the SSD gets written to the RAID array so that, should the SSD ever decide to stop being an SSD and become an expensive doorstop, that's okay, I will have an image current. And in fact I'm using a couple other utilities to capture file versions of the things I'm working on during the day so that I maintain essentially a nice versioning system.

But in response to my mentioning imaging last week, I got a bunch of people saying, what imaging software do you use? So I wanted to give a heads-up and props to my absolute favorite solution, which is called Image for Windows. I've mentioned these guys before, but not for years. The company is TeraByte, and the site is TeraByteUnlimited.com. And it's just a beautiful piece of work. It's inexpensive. It's less than $39. They've got a version for DOS and for Linux. It understands NTFS formats, so you're able to image a large drive into a single image file, rather than needing to be chopped up into two or four gigabyte files, which would be the case if you had a FAT formatted imaging drive.

It's available for 30 days with sort of a little timeout nag that comes up, but fully functional. Feature complete. You're able to expand and shrink and scale drives to the target. You can perform a drive-to-drive copy. You're able to image the drive to a single file. You can mount that file, if you want to, pull individual files out of images. You're able to do that, to perform a read-only mount.

Anyway, that's what I'm using. I've been using it for years and absolutely recommend

them without hesitation. And there's a bunch of other tools they have, too, a bunch that are free. Oh, it's also able to do an image of an in-use drive. It'll use the VSS service in Windows, the Volume Shadow Service, in order to allow you to copy files that are in use. But it also has its own PHYLock, they call it, that is their own kernel driver, if for whatever reason VSS is not available. And I think in some of the smaller versions, like Home or non-premium or Ultimate and so forth, I think maybe they don't offer the volume shadow service. PHYLock will essentially do the same thing. So anyway, love these guys: Image for Windows by TeraByte. And I recommend them without hesitation.

PADRE: Steve, you have the chatroom still kind of geeking out over the NVMe M.2 SSD. So eventually they'll catch up with everything else.

Steve: It's just this gorgeous, it's this gorgeous little thing.

PADRE: Isn't it nice?

Steve: It's like, oh, it's so cute. It's just - oh.

PADRE: I remember the first time I saw an M.2. It was the Kingston guys came over, and they showed me their new M.2. And I was looking at it going, this is it? I mean, really? This? But, yeah, actually they're starting to release the new - the four-lane. And the new PCI revision spec will actually double the amount of data that you can push over a four-lane PCIe. And at some point we're going to get faster than the PCI bus itself.

Steve: Well, and as it is now, you know how - because of course I'm using Windows 7. You know how Windows 7 comes up, and you get those four little sprites, or the four little twinkles that sort of spin around and then form the glowing window...

PADRE: They don't spin much anymore.

Steve: No. It boots so fast, they don't get themselves into position by the time the screen blanks and then the desktop comes up. So, and again, I don't boot my systems often, so it's not like I'm going to see that all the time. But it just feels good to have a nice, beefy, state-of-the-art machine. And I'm glad - I guess Intel or Microsoft did back off. So much heat was generated by their announcement that they weren't going to support, what is it, the xHCI, the predecessor to the EHCI USB controller, that that was going to cause people the inability to set up Windows 7 on newer hardware. And so that's what motivated me to say, okay, forget this. I'm happy with 7. I'm going to stay there, in the same way that I stayed with XP, for the next decade. And 7 is still officially supported through 2020, or until 2020. I think it's April something.

PADRE: I still have a Vista machine sitting somewhere, I think. I probably shouldn't have...

Steve: Necessary for testing.

PADRE: I probably shouldn't have admitted that.

Steve: So Leo and I also lamented the loss of AnyDVD.

PADRE: I use this tool.

Steve: Yup. I do, too. And I just, now, I have to say I'm using it less as the world has gone more connected, more Netflix and HBO Go and so forth. But back in the day, a disk

that I owned, I would often rip it and copy it to a laptop in order to watch when I was somewhere where it wasn't convenient to have DVDs. Or I would want to pull a segment off of it in order to keep, or whatever. The point was not piracy, but legal use of content that I had purchased.

And I love these guys. Every time I turned - like every week I turn on the system that I use just for Skype, essentially, and that's the one where AnyDVD is. And invariably I get a little pop-up saying, oh, a new version of AnyDVD is available. And of course they were doing this because it was the cat-and-mouse race of, okay, we're going to come up with a new way of encrypting our DVD that the rippers aren't going to be able to handle. And these AnyDVD guys would reverse engineer it and add a little bit of code and push out an update.

Now, they're back in the form of RedFox.bz. And I had somebody who sent me a note, to actually both me and Leo, saying, "Steve, you mentioned AnyDVD on Security Now! 549. If you have a license to any of SlySoft's products, you can still update them as the developers have resurrected the software. SlySoft, the company behind AnyDVD, has been closed down. But it appears the developers were not based in the same country (Antigua) as SlySoft. The new site is located here," and he gave me a link to forum.redfox.bz. Now, I went there because I was curious as I was putting the show notes together. And I did see a download in the upper right. Unfortunately, it looks like they've changed models. That is, the licenses apply up to the version one point release before the one they have now. And I looked around, but I didn't see a previous releases button. Maybe it's there somewhere.

But what's happened is they've switched to an online model. That is, rather than constantly pushing out updates, and then you download it, and you install it over the one you had, it's now you pull the latest from their server on the fly sort of model. And I don't know if it's per disk or how it's structured. Anyway, they have changed that. And frankly, I'm not going to bother re-upping my license or pursuing it. I haven't used the thing for years, as I mentioned. It's just sort of, you know, disks are not something that I need to be able to pull content from any longer, the way I once did. Everything's now connected. But still, I just want to - I did want to give people a heads-up because I know it was a very popular solution for people.

PADRE: The Jesuit community that I live in actually has a room as big as our regular book library that is filled with nothing but DVDs. And these are DVDs that have been bought by Jesuits over the last, as long as we've had DVDs. And so one of my tasks, one of my chores when I came into the house is I have gone through and consistently ripped all the DVDs to a 12-Bay 48TB NAS Array that we have in my data closet. And now everyone in the house has access to a menu system so they can just choose the DVD they want to watch.

Steve: Nice.

PADRE: And that's completely legal. We own the media. It's ours. And if anyone ever comes to check it out, we do have the room.

Steve: Here it is.

PADRE: There you go.

Steve: And besides, you're all Jesuits.

PADRE: Yeah. So, I mean, separation of church and piracy.

Steve: I did want to note one of the other sort of miscellaneous topics that this podcast has enjoyed entertaining from time to time is energy storage. I've been a big fan of the concept of a supercapacitor, the idea of using electrostatic storage rather than electrochemical storage. We've also talked about electromechanical storage, like maybe steam under pressure, or whirling dervish flywheels made of fibers anchored at the center, you know, and spinning in a vacuum magnetically suspended. The question is, how are we going to store energy in the future? And of course cars are wanting a portable source.

Just in the news is an interesting solid-state lithium technology that I just wanted to mention. This was the result of some research that Toyota and a group of academic researchers have been working on since at least 2011 because Toyota did some pioneering work and published a paper back then. What's interesting is that, when I say "solid-state," I mean actual solid. And it's weird to think of, like, a solid battery, that is, a battery that doesn't have some goo in it, because all the batteries we've ever known are goo-filled. And the goo is a problem because, as anyone knows who's like left old dry cells in a toy, you open it up, and it looks like some white fur farm has exploded in there because the goo leaked out of the old carbon zinc battery all over the place and, like, ruined the connectors and the electrodes and sometimes everything because it is highly corrosive.

Lithium-ion, or lithium-polymer, basically use a goo which has a problem in that it has a narrow temperature range. And there is a problem with thermal runaway, which all those using the hoverboards this Christmas were worried about after the coverage of several explosions of their lithium batteries because they did not have good battery manufacturing and battery management, charging management. But think about this: a solid battery, that is, like a brick. It's just so cool. So it has a wide temperature range. First of all, this is not a commercial product yet. You can't go get these. But it's very encouraging. A wide temperature range. The problem with goo is that it can overheat at the high end, and it can freeze at the low end. And in fact people don't realize that it is very bad for a lithium-ion or a lithium-polymer, a Li-Po battery, to be charged while hot. If mine is ever hot, I will wait until it cools off.

PADRE: Yeah. It'll actually break down the chemistry.

Steve: Yes, they do not like to be charged while they're hot. So these things, the new solid-state lithium, has an operating range - an operating range - from negative 30 to positive 100C.

PADRE: Okay, yeah.

Steve: So way below freezing to boiling, literally, temperature. It can also - and in the show notes, at the top, the first page of the show notes always has our Picture of the Week. In this case it's just a picture of this lithium lattice array. Basically they came up with a solution which is not a solution for the first time. It's a solid, which also offers extremely high charge and discharge rates, rivaling a supercapacitor. The thing that was so cool about a supercapacitor is because it was not chemical, it was electrostatic, you could theoretically drive your car into a charging station, where they had even bigger capacitors, and recharge your car in a matter of a minute or two. A lot of current would be having to go through that period of time. But if you could source that much current, the point would be something like a supercapacitor could accept a full charge in a short time. This solid-state lithium technology can do that.

Now, one of the other problems we have with lithium battery technology is cycle life. And this has good cycle life. Not as good as supercapacitors, which is essentially infinite cycle life unless it breaks down. After the first cycle they lose 10% of its first capacity. So charge it up, discharge it. When they charge it, the first recharge, it has lost 10%. Which sounds bad, except that it then survives the next 500 cycles, only losing an additional 15. So if you consider that first loss of 10, okay, that's your zero point. Then you're only losing 15%, I'm sorry, you're only losing, yeah, 15 percent over the next 500 cycles. And it can charge at the speed of a supercapacitor.

So not yet on the market, but nothing to leak. It is solid. Because it's solid they're able to get 20 times the lithium-ion content in the same space as a fluid because this matrix actually holds them and allows them to move through the crystal matrix. So I just wanted to bring it up as another interesting little bit of energy storage tech. Maybe this will be the one. Supercapacitors are neat. We're seeing all kinds of advances on many different fronts.

PADRE: Right. For me, it parallels the development of lithium graphene, lithium titanium dioxide cells. The advantage of those is they've increased surface area between the anode, the cathode, and the electrolytes, which allows you to charge them incredibly quickly without a huge performance penalty. But this, because it's solid, not only do you get increased surface access because of the 3D crystal lattice, but it also means you have a much higher energy density. So, yeah, that's - actually, I would love to see this technology combined with something like a lithium titanium dioxide. So a solid lithium titanium dioxide battery. I bet you could get outstanding charge rates.

Steve: And it feels like, I mean, imagine. I mean, it feels like alien technology. It's a brick. How can this brick, you know, it's like, I guess I'm just so used to a battery being goo, and folded sheets with stuff in it that, if you step on it, you know, it squirts out the end. And here it's a brick. It's like, how can this brick store energy? Anyway, very exciting.

Greg, my tech support guy, said that he was getting a lot of questions from people about SpinRite and SSDs. So he said, "Steve, I know that SpinRite fixes them, but I need some boilerplate that I can send." And I've written a lot of the boilerplate that Greg sends. It's wonderful. He makes my life possible. People tweet me with a question, and I say, "Please ask Greg. He'll take care of you." He answers email. He's 100% reliable. And that's why he told me last November he was celebrating his 20th anniversary working for GRC. Great guy. Anyway, he said, "Can I have some boilerplate about SpinRite and SSDs?" And so I wrote something that I thought I would just read into the record of the podcast. We've sort of talked around this a number of times, but I put it together sort of in one place sort of formally and liked the way it turned out.

So I wrote: "With solid-state mass storage, as with electromagnetic spinning mass storage drives, manufacturers have brought their costs way down and made their products competitive, mostly by cramming too much data into too little space, and in doing so they have traded away some of the reliability margin from their products. For the most part, today's spinning and solid-state drives work well. But as SpinRite's owners learn, it's never fun to become a statistic.

"When flash drives and SSDs began to penetrate the market, we had no idea whether SpinRite's days might be numbered. We knew that spinning drives would still be around for a long time, but it was certainly possible that SpinRite's recovery technology might have nothing to offer drives where nothing was spinning. Then we started receiving reports, and not just a few, from SpinRite's existing users, who had used it to successfully recover their solid state drives.

"It turns out that, in order to cram too many bits into too small a space, the same error correction codes (ECC) that were developed for spinning drives were employed for solid-state drives. The reason is the solid-state bits have been made so tiny that they are always on the verge of being unreadable, just like a spinning drive's bits. So the same technologies GRC developed years ago for reading those unreadable bits still work on today's and tomorrow's solid-state mass storage technologies. And that shows no sign of changing.

"However, unlike spinning disks, the process of changing any of a solid-state drive's bits - writing to the drive - eversoslightly fatigues those changed bits. They become just a tiny bit less reliable. Over a long period of time, solid-state drives can actually wear out and develop bad bits, which is another of the things SpinRite can deal with. But this 'write fatigue' means that SpinRite should only be used at Level 2, not Level 4 on solid state drives. Level 2 places SpinRite into a 'read mostly' mode, where it will rapidly scan the solid-state drive, looking for any trouble, and will only switch into recovery and testing mode (Level 4) in the spots where trouble actually exists. Though solid-state mass storage can develop new defects over time, the builtin controller, working in concert with SpinRite, is able to take bad spots out of service and keep the drives running well.

"So SpinRite is definitely useful for solid-state mass storage data recovery, repair, and maintenance; and it should be used at Level 2 on those drives."

PADRE: Now, this is something that I've found whenever I've used SpinRite to speed up my SSDs because I know that there's a very, very small charge because every flash cell is a capacitor. And that charge will weaken over time. So if I'm reading from the drive, occasionally it will be so low that it has to go at it a couple of times. It runs through its error-correcting routine to find out, was that on, or was that off? Whereas, when I run Level 2 of SpinRite, it will refresh that. So it'll look at it. And if it is low enough, it will rewrite to make sure that the right bit is in that cell.

Steve: Essentially recharging the capacitors.

PADRE: There we go.

Steve: Exactly.

PADRE: Yeah. It's funny because, just like your writer, I was wondering if SpinRite would go away once we all flipped over to flash. But I've found that SpinRite is now even more of a vital tool in my toolbox whenever I go to fix PCs.

Steve: Yeah. Well, and in fact that's what got me going with SpinRite 6.1 and a future v7 is, it's like, okay. As long as it's not dying, I'm not, either. So let's do a bunch of new things that people need.

PADRE: Right, right. Steve, I've got to say, I've read up about this exploit, and I'm really hoping that you can explain it to me because it sounds spectacular at the same time it sounds scary. What is DROWN?

Steve: So what I love about this is that it qualifies as a super clever hack. It caught people by surprise. It's a consequence of the long tails that existing cryptographic protocols have and how, as we were talking at the top of the show, even something that you're no longer using anymore, and therefore think can't bite you, still can. DROWN is a kind of an awkward acronym. It's one of those that was reverse-engineered. It's an

acronym or an abbreviation for Decrypting RSA with Obsolete and Weakened eNcryption, D-R-O-W-N. And they don't do the "E" of encryption, they do the "N" of encryption, in order to make DROWN, the abbreviation, work.

Okay. So here's what these guys found. First of all, anyone who wants more detailed information and to check if any server in question is vulnerable - for example, right now, Padre, check out VMware.com, one of everyone's favorite major VM suppliers. VMware.com isn't looking so good. TWiT.tv got fixed, apparently, so that's good. But anyway, the site is DrownAttack.com. That's the page these guys created as a placeholder for the attack. You can find their detailed paper, 22 pages of crypto stuff and the demographics and some charts and things. They also provide you with a form that you can fill in to check any server you like, to see what its current status is relative to the attack. And right now on the video feed, Father, you just brought up the VMware.com.

PADRE: That's VMware. Uh-oh.

Steve: And they've got a lot of big names...

PADRE: Ooh, that's not great.

Steve: ...that are vulnerable. Yeah.

PADRE: Ooh.

Steve: Not good. Okay. So here's what this is. Really clever. What these guys realized is a weakness in SSLv2, in the handshake of SSLv2, could be used to determine the private key of the server, and that could be used to decrypt state-of-the-art TLS communications. So let me sort of walk us through it. First of all - okay. First of all, nobody needs to worry about this. I mean, this is bad, but there's no known exploitation of this in the wild. This is at this point a theoretical attack. It's been demonstrated. We know it works. But people are already scrambling. OpenSSL has been updated so that it is impossible with the latest update to, as of a couple weeks ago, to configure a server which uses OpenSSL for its security, to configure a server to support TLS and be vulnerable to the DROWN attack. So these things are in the process of getting fixed.

Now, as I'll explain in a second, though, there are a huge number of servers with a version of OpenSSL more than one year old because there was a fix in OpenSSL that coincidentally fixed this problem, even though that wasn't its intent, a year ago. Yet a quarter of the servers that are on the 'Net are using, are still using that, a year worth of obsolete OpenSSL, and that has a problem which allows the DROWN attack to function much more quickly.

So, okay, first of all, what is it? The way it works is an attacker would passively record about a thousand connections between a client and a server. And this is significant because it means that any data recorded in the past - and we know that we've wondered if the NSA isn't using their massive amount of storage in Nevada for, like, just recording everything. Even if they can't decrypt it, they're sucking it all in on the off chance that, in the future, they will be able to. Well, this is an attack that is exactly of that kind, where, first of all, you need to record about a thousand connections between a client and the server that is under attack. Then the weakness is that, even though the client is not using SSLv2, many servers have not explicitly forbidden it.

For example, it may be way down the list of preferred ciphers so that the export-grade encryption, that's another requirement is that they still allow the reduced-strength,

export-grade, 40- or 56-bit encryption. And again, which they may have way down at the bottom of their list. But if it's not gone, if it's still there, then the operator of the server is probably thinking, well, all contemporary clients are going to be using TLS. Nobody's using SSL anymore. But, you know, we don't want to deny service to anybody. So we'll leave SSLv2 enabled. And this is assuming they're even thinking about it at all. If their version of OpenSSL is a year old, they probably aren't thinking about it at all. But SSLv2 is there. Export-grade, reduced-strength ciphers are available.

So what can happen is it is possible for the attacker to perform a large series of SSLv2 connection handshakes. It is in the connection handshake that the endpoints exchange their nonces, their list of supported ciphers. In this case the attacker would say, I want to use SSLv2, and I only understand RC4 40-bit export-grade security. And so the server would sort of shrug and go, wow, okay, you must be old, but fine. We have that. The problem is that, with enough of these SSLv2 handshakes, this uses the same concept I mentioned before of an oracle. This turns the server into an oracle, meaning that you're able to essentially ask it for some secrets, or you're able to ask it in a way that causes it to reveal a secret.

And in this case, using a crazy amount of sophisticated math, the use of a high number of these SSLv2 handshakes at the lower encryption strength level allows the attacker to gain information to allow them to decrypt a TLS connection. They don't get the private key, so that's important. This is a session attack. So the private key is not compromised. But they are able to get enough information to decrypt the contents of one of the previously recorded sessions.

So in terms of severity, what these guys did, they found two attacks. They have what's know, what they call the "slow attack," which was the first thing that they developed, before they realized there a problem in older versions of OpenSSL that would allow them to speed it up a lot.

So the slower, more general attack, as they wrote, exploits a combination of unnoticed protocol flaws in the v2 of SSL, which allows them to create a stronger version of a well-known - it's called the Bleichenbacher attack, which has been known to cryptographers for a long time. It is an oracle-based attack that is a way of getting servers to disclose information. Yet there are countermeasures in all modern servers, specifically to thwart this. Well, it turns out that the countermeasures can be leveraged to create a stronger attack. So in the slow version of the attack they wrote that: "A typical scenario requires the attacker to observe a thousand TLS handshakes, then initiate 40,000 SSLv2 connections and perform 2^50 offline work," which is a lot of work, which would then allow them to decrypt a 2,048-bit RSA-encrypted TLS ciphertext. So that's why I say, okay, theoretical, kind of.

PADRE: Do they have to do anything different with those 40,000 individual SSLv2 connections? Are they incrementing something?

Steve: What they're doing is they are - it's a dynamic attack. So they are deliberately changing the behavior of what they ask for based on what the results are. So it is an adaptive attack.

PADRE: Okay. Got it, got it.

Steve: Essentially. Okay. So then they said: "An implementation of the attack [that they created] can decrypt a TLS 1.2 handshake" - so like a state-of-the-art, the best crypto we've got - "using 2,048bit RSA in under eight hours using Amazon's EC2 [the elastic cloud computing] at a cost of $440." Which is regarded as feasible, although it takes

eight hours of computing. That's that 2^50 offline work, after collecting all of this data. And in their Internetwide scans they found 33% of all HTTPS servers are vulnerable to this protocollevel attack due to widespread key and certificate reuse.

Okay, now, that the good news in terms of this attack being slow and difficult. They also uncovered on OpenSSL older than a year a fast version. That version allows essentially real-time decryption. They were able to, as quickly as they collected data and processed it on a single-core laptop, they were able to decrypt the connection, essentially in real-time, meaning that in a minute before the SSL protocol expired, before the TLS handshake gave up, they were able to get the connection decrypted, making a man-in-the-middle attack, a real-time man-in-the-middle attack, feasible. And 26% of HTTPS servers are still vulnerable to this attack now, one year after OpenSSL was upgraded, not with this in mind, to fix something else. But it also prevented this fast version of the attack.

PADRE: And that's with the minimal amount of offline work required.

Steve: Yeah.

PADRE: Basically no offline work required.

Steve: Exactly. You could be in real-time, it is so much faster using those older versions, where one quarter of the systems are vulnerable. Now, here's the other cool thing about this, and that is that, say that your HTTPS server was not using SSLv2, that is, you were properly configuring this, and when it got deprecated, you shut it down. But you also had email. And email is like, oh, it's email, who cares about email? But because certificates are a pain to get, and they're not free...

PADRE: We reuse them.

Steve: ...you reused the same certificate to protect your email server, which uses v2, because who cares about email really? And you use the same certificate. It turns out that this is completely vulnerable. This attack works on different servers using the same certificate. So even if you've got your web server bolted down, you're now using only TLS protocol, the state-of-the-art best protocol we've got, if you ever, anywhere - and not only email, any SSL. So you might have some funky who knows what, you know, particular corporate, you know, you might have a VPN where you've reused that same certificate, you know, a TLS VPN that also runs SSL so that it's able to operate with older clients. Any reuse of the same certificate on any other server allows this attack.

So if you cannot make sure that you are not offering the main cert on any other server that allows SSLv2 connections, you absolutely want to rekey one of the certificates, probably just get, you know, rekey the server you really care about that is not running any older protocols. And that way the older protocol-based servers cannot be used to acquire data on the connections that the state-of-the-art protocol server is using, and then you'd be safe. But I just love this. This is just such, as I said, a cool hack. The idea that you could use a deprecated protocol, still lingering because, okay, clients aren't asking for it, so it's not going to hurt us because all clients are going to ask for newer protocols. And here's the reason why you don't want the old ones.

And of course politically the academics are saying, "And see? This is what happens when the government forces us to reduce the strength of crypto." Back in the '90s, in order to export this "munition," we had to reduce the strength to 40 bits. That's biting us decades later and reducing the strength of crypto. Yet another perfect example of why it's never a

good idea to deliberately use weakened crypto. And as Matthew said in his blog post earlier, it's hard enough, maybe dauntingly hard, to get crypto right. The last thing you want to do is to deliberately make it less right.

PADRE: And that's why I have a lot of my Interop and Black Hat friends who they're spending a lot of their time looking for deprecated modules. They spend a lot of their time looking at OPSEC, saying these servers have been up a really, really long time. And you may have patched them, but you probably don't know what's behind every port. I mean, very few system administrators actually go to the trouble of making sure that everything has been deactivated after they've patched. And now it's time for them to do that. And that's - I'm with you. This is a cute hack. This is a unique hack.

Steve: Oh, it's wonderful.

PADRE: It's not just you hammering on a system. This is you counting on a part of human nature which is, once I've installed the patch, I assume everything's okay.

Steve: Yeah. Or you might have some server in the closet that you don't care about; but just because certificates are not free, you use the same certificate on multiple servers, and you don't think that your IoT server is going to be a problem. But, uh, whoops.

PADRE: And once I compromise that certificate, everything is fair game.

Steve: Yup.

PADRE: Steve, when is the security bamboozle ever going to stop?

Steve: No. We're at Episode 553 and, what, in our 11th year of the podcast. I can see this thing another 10 years downstream.

PADRE: Well, folks, you have used up 2.5 hours listening to...

Steve: We broke a record.

PADRE: A record.

Steve: I knew it was going to happen because we had a lot to talk about, and you and I tend to engage. So it's been great, Padre. It's really, really good to have you on the podcast.

PADRE: We could have spent 2.5 hours just talking about those first two stories. I mean, those are very, very juicy stories. But I'm glad we got to DROWN because, after you sent me that link, I was just looking at every site that I've ever been part of, just making sure that they're patched. And I've had to send out quite a few emails saying, "Hey, so, um, listen. If you go to the site, check your security. You've got a couple of servers that need some work."

Steve: Yup.

PADRE: Of course Steve Gibson is the man behind GRC.com, where you'll find tools like my personal favorite, SpinRite, as well as ShieldsUP! and, soon, I know what's going to happen when I ask this, but the savior of our password race, SQRL.

Steve: Yup.

PADRE: Should I ask about that?

Steve: Well, yeah. Everybody knows I am at work on it. I am in the process of updating the client to the full final spec. I wrote the spec in advance of getting the client caught up because we've got other people who are implementing SQRL clients, too. Jeff Arthur, who's our friend over in the U.K., he's already got his iOS client ready, but he's just waiting for me to get my client going because this changes an aspect of the identity so that the clients will be able to remember four retired keys for an identity. I'd always had it able to remember THE retired key, that is, the most recently retired key. In theory, you should never need to retire a key. But in practice we know, as they say, stuff happens.

And so we needed to have a provision, and did from the beginning, where if somehow your phone got hacked, or you suspected it was, or the government had control of it overnight, and for whatever reason you could no longer trust your SQRL identity's key, it was always possible to rekey. SQRL would then contain both the old and the new simultaneously. So then all you do is go visit sites that know you under the old key, and they update autonomously. You have to do nothing. You just have to go there, and it automatically updates their knowledge to your new key.

Well, the concern was, okay, but what about a site I don't visit for a long time, I forget about it, and I rekey my rekey? And so it's like, okay. For a while I fought it, and I said, you know, come on. And then finally I gave up. And so I decided, okay. Rekeying should be something most people never do, but it's feasible. And I don't want people to have to, like, keep a log or a diary of all the sites that know them. So I decided, okay, we'll let a SQRL identity carry the four previous retired obsolete keys. And so when you go to a site that doesn't recognize you, but it should, it'll be able to check back four times, through four previous keyings, in order to see if it can find you. If it does, it immediately knows, it absolutely knows who you are because all of this is secure and can't be, as far as anyone knows, messed with. It'll then essentially ratchet to your most recently issued identity and be kept synchronized.

Anyway, I'm in the process right now of updating my client to support the four previous keys, rather than the one previous key. And at that point, as far as anyone knows, it's done. I have some stuff to clean up in the client, but the protocol, the spec is in place. We've got a bunch of people working on Android versions. We've got one already for iOS. So we're getting up to speed.

PADRE: If you need any beta testers, you've got a chatroom full of people who would willingly demonstrate their devices.

Steve: Well, and we do - I've not published it, but GRC.com/sqrl/demo.htm is the server that is producing, I mean, it's an operating SQRL demo server, which those of us who are testing the stuff now are using. So, in fact, that's what I had Leo log onto many months ago. So we're close.

PADRE: Well, Steve, I hope that Leo goes on plenty of vacations this year because I always have so much fun. Whenever he gets to go away, I get to sit in this chair; I get to sit at this table and speak with you. And it's the best time ever. So thank you very much. It's been an honor and a privilege for you to dump your brain into mine. I feel smarter, and I think the Internet feels smarter, too.

Steve: Well, you've helped create another great podcast, Padre. Thanks very much.

PADRE: Thank you.

Steve: Next time.

PADRE: Don't forget that you can find Security Now! here on the TWiT.tv network. It's recorded live every Tuesday at 1:30 p.m. Pacific time. You can also go to GRC.com for audio versions of the podcast, as well as wherever fine podcasts are aggregated. Again, Steve will be here with Leo Laporte next week. Until then, I'm Father Robert Ballecer, the Digital Jesuit, just reminding you that, in a world of exploits, we all need Security Now!.